

AD-783 626

THE CAMBRIDGE PROJECT: COMPUTER
METHODS FOR ANALYSIS AND MODELING
OF COMPLEX SYSTEMS

D. B. Yntema

Massachusetts Institute of Technology

Prepared for:

Rome Air Development Center
Advanced Research Projects Agency

February 1974

DISTRIBUTED BY:

NTIS

National Technical Information Service
U. S. DEPARTMENT OF COMMERCE
5285 Port Royal Road, Springfield Va. 22151

ACCESSION NO.	
BY:	WHILE SERVING <input checked="" type="checkbox"/>
OR:	POST SERVICE <input type="checkbox"/>
REMARKS:	
BY:	
REMARKS:	
DATE: APR 10 1964	
<div style="border: 1px solid black; padding: 5px; display: inline-block;"> A </div>	

If this copy is not needed, return to RADC (IRDA) S. DiCarlo, GAFB, NY 13441

AD 783 626

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-74-159	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) The Cambridge Project: Computer Methods for Analysis and Modeling of Complex Systems		5. TYPE OF REPORT & PERIOD COVERED Technical Report Interim July - December 1973
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) D. B. Yntema and Cambridge Project Participants		8. CONTRACT OR GRANT NUMBER(s) F30602-72-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS Massachusetts Institute of Technology 575 Technology Square Cambridge, Massachusetts 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61101E, ARPA Order 1756 17560101
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Blvd Arlington VA 22209		12. REPORT DATE February 1974
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) RADC/IRDA ATTN: Samuel DiCarlo Griffiss AFB NY 13441		13. NUMBER OF PAGES
		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release. Distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Approved for public release. Distribution unlimited.		
18. SUPPLEMENTARY NOTES Reproduced by NATIONAL TECHNICAL INFORMATION SERVICE U S Department of Commerce Springfield VA 22151		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Computer programming Time-series Analysis Behavioral Science Computer-based Modeling Statistical Analysis Systems Analysis		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The Cambridge Project is a cooperative effort by a number of scientists at M.I.T. and Harvard; its purpose is to make the digital computer more useful and useable by scientists in the basic and applied behavioral sciences, and in other sciences that have similar computing problems. This Technical Report describes progress during the half year beginning in July 1973. The Project is supported by the Advanced Research Projects Agency under Contract F30602-72-C-0001.		

36

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

The Project has two main goals: first, developing programs and other computing tools; second, combining these tools, and tools borrowed from other sources, into a Consistent System of programs, models, and data that behavioral scientists can use on-line. Most of the effort is devoted to writing and documenting programs, but attention has been paid to theoretical studies of statistical techniques, equipping a computer-based laboratory for studying autonomic conditioning, and other related subjects.

This report describes work toward achieving the second of those goals, the Consistent System, which has increasingly become the focus of the Project's attention as time has passed, work toward accomplishing the first goal, the development of computing tools; tools for data-handling; tools for data-analysis and modeling; and computer-controlled experiments and studies of human factors in on-line computation.

The most notable single achievement of the half year covered here was the transfer of the entire Consistent System from the old Multics computer at M.I.T. which was a Honeywell 6180, and the subsequent transfer to another 6180. In spite of the inevitable differences in operating system that were encountered, both of these transfers went smoothly. This was in large part due to the design of the Consistent System, a design that was intended to isolate the collection of programs from the details of the time-sharing system in which they run.

ja

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

THE CAMBRIDGE PROJECT: COMPUTER METHODS
FOR ANALYSIS AND MODELING OF COMPLEX SYSTEMS

D. B. Yntema
and
Cambridge Project Participants

Contractor: Massachusetts Institute of Technology
Contract Number: F30602-72-C-0001
Effective Date of Contract: 15 July 1971
Contract Expiration Date: 15 July 1974
Amount of Contract: \$4,301,311.00
Program Code Number: 3D20

Principal Investigator: Joseph C. R. Licklider
Phone: 617-253-7705


Project Engineer: Samuel S. DiCarlo
Phone: 315-330-3126

Approved for public release;
distribution unlimited.

This research was supported by the
Defense Advanced Research Projects
Agency of the Department of Defense
and was monitored by Samuel DiCarlo
RADC (IRDA), GAFB, NY 13441 under
Contract F30602-72-C-0001, Job Order
Number 17560101.

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS).

This technical report has been reviewed and is approved.


RADC Project Engineer

ABSTRACT

The Cambridge Project is a cooperative effort by a number of scientists at M.I.T. and Harvard; its purpose is to make the digital computer more useful and usable by scientists in the basic and applied behavioral sciences, and in other sciences that have similar computing problems. This Technical Report describes progress during the half year beginning in July 1973. The Project is supported by the Advanced Research Projects Agency under Contract F30602-72-C-0001.

The most notable single achievement of the half year covered here was the transfer of the entire Consistent System from the old Multics computer, which was a Honeywell 645, to a new Multics computer, a Honeywell 6180, and the subsequent transfer to another 6180 operated by the Air Force Data Services Center. In spite of the inevitable differences in operating systems that were encountered, both of these transfers went smoothly. This was in large part due to the planning of the design of the Consistent System. This design was intended to isolate the collection of programs from the details of the time-sharing system in which they run.

TABLE OF CONTENTS

1.0	SUMMARY		1
2.0	THE CONSISTENT SYSTEM		3
2.1	The Present Collection		3
2.2	Documentation	Kay Gaudreau Caroline Lange Caroline Tompkins	5
2.3	Utility Subroutines and Support Programs	John C. Klensin	9
2.4	Substrate and Preprocessor	Robert Sorrentino	8
2.5	Conventions and Acceptance Procedures	Susan Godsell John C. Klensin Jeffrey Stamen Douwe B. Yntema	9
2.6	Contacts with Users	Martin Broekhuysen	9
3.0	DATA HANDLING		13
3.1	Janus	Jeffrey Stamen Robert Wallace	13
3.2	SURVEIR Data Management Project	Michael Ross Wendy Mela	13
4.0	DATA ANALYSIS		15
4.1	Project Diana	Rosemarie Rogers Elaine Franklin E. McCabe	15
4.2	TSP Project	John Brode Paul Werbos	16
4.3	DATATRAN	John Brode Jeffrey Stamen Robert Wallace	19

Preceding page blank

4.4	Economic Application of Spectral Analysis	R. F. Engle	20
5.0	MODELING		23
5.1	The General Implicator	Ithiel de Sola Pool Shahriar Ahy	23
5.2	Discourse	W. McMains	23
6.0	HUMAN FACTORS		27
6.1	A Modular Computer System for the Study of Autonomic Behavior	Craig Fields	27
6.2	Simulation Gaming as a Behavioral Laboratory	P. G. W. Keen	27
6.3	IMLAC Graphics	Nicholas Negroponte M. S. Miller	28

PARTICIPANTS IN THE CAMBRIDGE PROJECT

Jul-Dec 1973

Ahy, S.
 Akemann, R. A.
 Aron, J. L.
 Bartlett, L. C.
 Berkey, D. A.
 Blaydon, C. C.
 Bolt, R. A.
 Brode, J.
 Broekhuysen, M. J.
 Cartagena, C.
 Chan, A.
 Czerwinski, A.
 Davis, E. W.
 Dempster, A. P.
 Deutsch, K. W.
 Dinur, D. D.
 Dixon, G. B.
 Engle, R. F., III
 Fields, C.
 Fleisher, A.
 Franklin, E. C.
 Gaudreau, A. K.
 Gerstmyer, R.
 Gilbert, J. P.
 Gilson, C. R.
 Glauber, R. R.
 Godsell, S. W.
 Griffel, D.
 Griffith, W. E.
 Hauck, W.
 Hill, P. S.
 Keen, P. G. W.
 Klensin, J. C.
 Krieger, A.
 Lam, Chat-Yu
 Lange, C. M.
 Lee, K. C.
 Libby, C.
 Licklider, J. C. R.
 Lucht, L.
 Magnussen, N. J.
 Mammano, D.
 Markowitz, J.
 McCabe, E.

McKenney, J. L.
 McMains, W. M.
 Mela, W.
 Miller, M.
 Negroponte, N. P.
 Newman, E. B.
 Newmark, E.
 Neyer, M.
 Nichols, P. M.
 Norton, J.
 O'Connell, K. M.
 Olivier, D. C.
 Palter, G.
 Pettigrew, T. F.
 Pool, I. de Sola
 Porter, W.
 Rackley, K.
 Rogers, R.
 Roos, D.
 Ross, J. M.
 Rothrock, L. S.
 Rubin, D. B.
 Schachter, R.
 Schlaifer, R.
 Schleifer, A., Jr.
 Scott, R. H.
 Selfridge, O. G.
 Shields, R.
 Shuford, D. F.
 Siegel, M.
 Silverman, R. W.
 Snyder, D.
 Sorrentino, R. K.
 Stamen, J. P.
 Strayhorn, J. M.
 Sussman, J.
 Taylor, J.
 Tompkins, C.
 Treimanis, N.
 Tudor, G. G.
 Wallace, R. M.
 Weiler, S.
 Werbos, P.
 White, J.
 Yntema, D. B.

1.0 SUMMARY

The Cambridge Project is a cooperative effort by a number of scientists at M.I.T. and Harvard; its purpose is to make the digital computer more useful and usable by scientists in the basic and applied behavioral sciences, and in other sciences that have similar computing problems. This Technical Report describes progress during the half year beginning in July 1973. The Project is supported by the Advanced Research Projects Agency under Contract F30602-72-C-0001.

The Project has two main goals: first, developing programs and other computing tools; second, combining these tools, and others borrowed from other sources, into a Consistent System of programs, models, and data that behavioral scientists can use on-line. Most of the effort is devoted to writing and documenting programs, but attention has been paid to theoretical studies of statistical techniques, equipping a computer-based laboratory for studying autonomic conditioning, and other related subjects.

Section 2 of this report describes work toward the second of those goals, the Consistent System, which has increasingly become the focus of the Project's attention as time has passed. The remaining sections report on work toward the other goal, the development of computing tools: Section 3 is on tools for data-handling; Section 4, for data-analysis; Section 5, modeling; and Section 6, for computer-controlled experiments and studies of human factors in on-line computation.

The most notable single achievement of the half year covered here was the transfer of the entire Consistent System from the old Multics computer at M.I.T., which was a Honeywell 645, to a new Multics computer at M.I.T., a Honeywell 6180, and the subsequent transfer to another 6180. In spite of the inevitable differences in operating systems that were encountered, both of these transfers went smoothly. This was in large part due to the design of the Consistent System, a design that was intended to isolate the collection of programs from the details of the time-sharing system in which they run.

Work will continue during the next half year under the same contract.

2.0 THE CONSISTENT SYSTEM

The project has continued to focus more and more on the Consistent System. In fact, most of the work described in the other sections of this report is on programs that are already components of the System, or are intended to become components of it.

The System is a collection of programs that are intended for interactive use, and that are, moreover, designed so that a scientist who is not a programmer can use them together. It runs within the Multics time-sharing system, and is expected to become a repository not only of programs, but of models and sets of data, and it appears potentially applicable to uses far wider than the Behavioral and Social Sciences, for whose needs it was originally designed.

The collection now consists of about 200 programs - some quite small and some that are whole systems in themselves - which add up to almost a million words of object code. Early in the period covered by this report the collection was successfully transferred from the old Multics computer at M.I.T., a Honeywell 645, to the new one, a Honeywell 6180. Shortly thereafter the entire collection was also transferred to the 6180 operated by the Air Force Data Services Center.

The report on the period from July 1972 to January 1973 gave - on pages 2-1 to 2-22 - an extensive account of the System, including its general organization, its background and goals, and the kind of consistency at which it aims.

2.1 The Present Collection

Of the 200 programs now in the collection, about 160 are of interest to the user who is making serious use of the System today; the others are waiting for companion programs that will, for example, make it easy for him to prepare inputs for them or display their outputs. These 160 programs divide, from his point of view, into roughly four groups:

(1) The prototype of Janus. This is a large system that handles data of a kind common in fields like behavioral science. A typical set of data contains information about a number of "entities" (e.g., people), each of which has a number of "attributes" (e.g., age, sex, years of schooling, occupation, town of residence, and so on). Janus has some unusual powers, many of which come from its ability to handle relations between

such sets of data. For example, if there is a set in which the entities are people, and another in which the entities are towns, a relation between the two sets can be used in computations that depend upon the attributes of the town in which each person lives.

(2) TSP-CSP. This is a large system originally designed for batch processing. Its full name is "Time Series Processor - Cross Section Processor". While originally intended for econometricians, it has a great deal of statistical power that is useful to investigators in other fields. It includes:

- ordinary least squares (i.e., multiple linear regression)
- weighted least squares
- least squares with instrumental variables
- residual analysis
- extrapolation or forecast analysis
- nonlinear least squares
- Bayesian regression
- regression with autocorrelated errors
- spectral analysis
- principal components
- factor analyzer (varimax method)
- correlation analysis
- algebraic operations on vectors and scalars
- matrix arithmetic
- scatterplots and plots of time-series.

(3) Small programs that work on numerical arrays. There is a collection of about 90 programs that accept or produce multidimensional, rectangular arrays of numbers. This part of the collection offers the investigator who is not a programmer great flexibility in numerical calculations; by using sequences of these programs he can perform computations for which he does not find convenient provisions elsewhere in the collection. About half of these programs are statistical. Others do matrix arithmetic, extract and replace subarrays, plot graphs on a CRT or typewriter terminal, and so on.

(4) Miscellaneous. This category includes: (a) programs that permit the nonprogrammer to define a "macro-command" - i.e., an agent that will run off a sequence of programs for him; (b) doorways to other systems that run within Multics but have not been maintained by the Cambridge Project - e.g., doorways to APL and to text editors; (c) service programs that permit the user to list the files he has in storage, delete files he no longer wants, gain access to another user's files (if the other user has told Multics to permit him to), leave the Consistent System, and so on.

From the point of view of the nonprogrammer, one of the most important aspects of this collection of programs is their compatibility. Janus can accept arrays of numbers from TSP, TSP can accept them from the array programs, which can accept them from Janus, and so on. Thus the investigator can use the programs together in a way that lets their capabilities reinforce each other.

The project is at present devoting most of its effort to expanding and improving this collection. At the risk of repeating information given in other sections of this report, the main areas into which the effort is going are: a new Janus that will replace the present prototype; a new language, tentatively named "DATATRAN", through which the user may communicate with the new Janus, the routines now in TSP, and other statistical programs now in preparation; a collection of programs for studying passages of natural text; an improved set of programs for graphics on CRT terminals; and programs for modeling of several kinds.

2.2 Documentation

Principals: Caroline Lange, Caroline Thompkins, A. K. Gaudreau

At least one of the members of the Project's faculty Advisory Committee is of the opinion that the Consistent System is the best-documented large body of software he has seen. In any case, considering that this is a system still under development, the thoroughness of the documentation can justly be regarded as unusual. While sheer volume is no guarantee of quality, it tells at least part of the story: there are now about 1850 pages of documentation, 1000 for nonprogrammers, and 850 for those programmers who want to add new programs to the collection. Those figures do not include the additional, archival documentation that is kept on file for later use by programmers who need to maintain or modify existing programs or the Substrate on which the collection rests.

An important addition to documentation for users has recently been made: a draft of an introductory manual for nonprogrammers who want to use the Consistent System has been prepared by Caroline Lange, and it is being circulated on a trial basis.

The cornerstone of the remaining documentation for users is

the Handbook of Programs and Data, which contains a description of every program in the public libraries - both the "a" library, which contains programs that have been formally accepted into the collection, and the "x" library, which contains programs that are still in an experimental status. Normally, this Handbook attempts to describe the program thoroughly from the point of view of a nonprogrammer, telling him everything about it that he might ever want to know and perhaps a little more besides. The main exceptions are two programs that are systems in themselves, TSP and the prototype Janus: each of them has its own user's manual.

An important addition has also been made to the documentation for programmers who want to contribute programs to the collection: a draft of a manual for contributors was prepared by John Klensin during the fall and is being circulated on an experimental basis. That manual serves as an introduction to the fundamental document for programmers, namely the Handbook for Programmers which describes from the contributor's point of view the Substrate on which the collection rests; states the conventions that contributions to the collection must obey, and the additional rules they should obey to be considered "regular"; describes the file Description Schemes that have been officially accepted; describes the "public subroutines" that a program may invoke; and gives the standards for documentation of public programs and subroutines. This reference book has grown considerably during the past half year, mainly through the introduction of the descriptions of about 40 new utility subroutines.

To summarize activities in documentation: the Handbooks are updated periodically; papers have been written and presented; and, several tutorial documents have been published. The Substrate Logic Manual, mentioned in the last report, has been published as: "The Multics Substrate Implementation", Klensin, John C., The Cambridge Project, M.I.T. Dec. 1973. Also published as a preliminary version was "The Beginner's Manual for the Consistent System", Lange, Caroline S., The Cambridge Project, M.I.T., November 1973. This primer contains an introduction to the Consistent System for the new, nonprogramming user: it defines the elementary concepts; explains how to enter and leave the CS, discusses the user/terminal environment; explains how to enter data; describes the service commands; and, how to create and run 'macros'. A more technical manual for programmers was also published, written by John C. Klensin, and entitled "Programming The Consistent System".

The following papers and reports have been published and/or presented during this report period:

1. "Janus: A Data Management and Analysis System for the Behavioral Sciences", Jeffrey Stamen and Robert Wallace, which was released after its presentation at the August ACM meeting in Atlanta, Georgia.
2. "ANSI TAPE Support on Multics", Godsell, S., The Cambridge Project, M.I.T. 4 Sept. 1973.
3. "An On-Line Management System for the Secondary Analysis of Public Opinion Data", Ross, J. Michael and Mela, Wendy; paper presented at the Ninth Annual EDUCOM Conference, Princeton, N.J., 10 October 1973.

2.3 Utility Subroutines and Support Programs

Principal: John C. Klensin

During the reporting period, work has been completed on the utility subroutines discussed in the last Technical Report. These routines provide convenient ways to access and create files and to analyze input directions for particular programs. In addition to those designed during the last reporting period for which coding has been completed during this one, a complete set of utilities for the labeled array description scheme "genarray" has been defined, coded, and documented. This description scheme will be the basis for most of the numerical and statistical programming to be done during the balance of the project. The programs designed to deal with it, in addition to handling labeled data, are also somewhat more flexible than earlier routines in the way they handle arguments and return values.

In addition, a set of utility subroutines has been written and documented to provide easier handling of unformatted character files of description scheme "char". These utilities permit acquiring these files, concatenating them, making simple character strings from them, and so forth, without requiring the programmer to have a detailed knowledge of Consistent System file handling. Similar routines for numerical files are included among the "genarray" set.

Besides the utilities for ordinary programs, a collection of utilities has been specified, and largely coded, to facilitate the construction of Consistent System agent programs. These

utilities include a few routines for interpreting commands and running other programs, and an integrated collection of routines for dealing with errors, passing the information about the errors to the user, and responding to his requests with respect to errors that have already occurred.

With the completion of the error and agent utilities during the first portion of this reporting period, the simple macro runner originally constructed for the system two years ago has been extensively revised. This program provides a user the capability of constructing a new command from a series of individual commands without having to resort to an ordinary programming language to do so. The runner permits specifications of command names and arguments in the same fashion as the "exec-com" or "runoff" facilities of many time-sharing systems. In addition, it provides, through special control statements, facilities for conditional and unconditional iteration, conditional branching, and similar programming language-like facilities. It also has provisions for creating formal parameters and local temporaries using the name translation facilities of the Consistent System, and provision for sending output to or receiving input from selected files, rather than the terminal. The design and most of the coding of this routine has been completed. Documentation and final testing is in progress and should be completed during the next few months.

This simple macro runner provides what is basically a procedure-oriented technique for dealing with a series of commands. One lists the commands to be executed, one per line, with whatever control statements are needed to control the execution. A second macro-like facility has been designed for handling nested groups of commands as if they were a single command execution. The basic utilities and preliminary documentation for this facility are complete, and it is expected that the facility itself, tentatively called the "fn" runner, will be completed during the next half year.

2.4 Substrate and Preprocessor

Principal: Robert Sorrentino

The Substrate, on which the balance of the Consistent System depends, continues to function smoothly, with no bugs having been found by users during the reporting period. A Substrate Logic Manual describing completely the internal interactions of the Substrate on a module by module basis, was released early in the reporting period and is available to those who are interested in it.

The Consistent System was moved to the Multics follow-on machine during July with little difficulty, due to the stability of the Substrate and preparations done during the preceding three or four months. Later the System was moved, with even less difficulty, to the Multics at the Air Force Data Services Center. The fact that both moves went smoothly in spite of inevitable differences in operating systems, and the fact that the System has continued to work in spite of inevitable changes in the M.I.T. Multics, are regarded as further evidence of the value of a Substrate that isolates the collection of programs from changes in the underlying software.

Other work on the Substrate during this period consisted mostly of extensive timings of Consistent System behavior and the beginning of redesign and recoding of critical portions of the Substrate to take better advantage of hardware features of the new machine. This process has been somewhat delayed by the fact that these facilities have been only gradually turned on, in spite of the fact that the user transition to that machine was made in July. The process of recoding and testing is expected to continue into the spring, after which a new and final edition of the Substrate Logic Manual will be produced.

Work continues on the preprocessor, but it has been further delayed by the press of work on the Substrate, the new machine, and similar problems. Its completion is expected during the first part of the next half year.

2.5 Conventions and Acceptance Procedures

Principals: Susan Godsell, John C. Klensin,
Jeffrey Stamen, and Douwe B. Yntema

The procedures for accepting programs and utilities into the System continue to work smoothly - so smoothly, in fact, as to be little noticed by much of the Project. The procedure continues to be much as described in the previous annual report, except that with departure of Jeffrey Stamen from the Project Staff in the fall, Susan Godsell has become a member of the acceptance committee.

The library continues to grow with more than 400 programs and subroutines now available in the system.

2.6 Contacts with Users

Principal: Martin Broekhuysen

In the period July through October we helped with the Multics registration of six new projects that came to Multics with the primary purpose of using the Consistent System. We know that other persons and projects began in this period to use the CS; we estimate several dozen. The six included five who work out of Harvard's William James Hall (behavioral-science center): four faculty plus the user-consultant group, all of them with considerable experience in computerized data analysis; and one faculty member of Harvard's School of Architecture, with little previous experience. Each of these was given as much of the Project's user documentation as we determined they could profitably use, and more or less time in initial consultation - ranging from an hour or so up to a day - spent on some particular problem such as retrieving the contents of a previously-existing directory or entering data not initially on punched cards. After an initial period of two weeks or so in which they might have several questions in up to half a dozen phone calls, none of these have raised significant problems. We do know that they are continuing to use the system -- are logging in fairly regularly. Also, the total number of users is considerably larger than this group. We hear by telephone from persons who were using the system before that period and still are; we have members of the Central Staff in frequent contact with the users in Washington; in the Executive Office of Manpower Affairs of the Commonwealth of Massachusetts; at the University of Illinois; and, users enrolled in a fall term course in the Department of Urban Studies at M.I.T. On Janus alone, there is currently an average of about five logins per day. Judging from the number of persons known to use the CS regularly, we can say that there are many who are getting along on their current knowledge and the user documentation, without needing face-to-face consultation or other assistance.

We know enough about many of them to identify some research interests of the group. Three areas are represented by three or more users: medical-data analysis; budget management and resource-allocation problems (Air Force Data Service, the Commonwealth, the urban-studies course); and sociological

survey data (the new Harvard users). Within the Project, Janus has proved an effective management information system for the control of Project documentation.

Regarding users' experience with the system as a whole, we have not, even when we asked specifically for global criticisms, received really global complaints on the order of "too costly to use" or "can't find any way around the system (or documentation)". Several persons have returned with quite detailed lists of suggestions for changes or additions to the documentation, some suggestions for changes in the system, and infrequent reports of program bugs. Users asking for advice are usually in the position of needing more information than they already have, about a specific problem solution, and are seldom in the position of claiming that something that they want to do can't be done.

About the utilization of the system facilities: we know there is very substantial use of Janus, and considerable satisfaction with its capabilities (in particular with the dataset functions), as well as needs for many of the powers that will be available in the full Janus but are not in the prototype. At least one user is doing considerable imaginative work with the Reckoner programs in the manner in which they were intended, i.e., combining them with Janus in such a way as to implement more sophisticated analyses (Bayesian probability procedures). At least one user is not doing data analysis but is finding the CS a convenient entry to APL where he is testing programs of his own for new statistical analysis.

Our experience to date suggests that the Consistent System is well regarded among its current users; more extensive usage will provide reinforcement to this belief. So far we have solicited only a limited set of users and have not broadcast the availability of the system in the computing community. The next six months should prove to be a more fruitful period for gathering concrete data from a larger group of users.

3.0 DATA HANDLING

3.1 Janus

Principals: Jeffrey P. Stamen, Robert M. Wallace

Contributors: Pamela Hill, Chat-Yu Lam,
Gary Palter, Dorothy F. Shuford, Marcia Siegel

The Janus system is described in the paper "Janus: A Data Management and Analysis System for the Behavioral Sciences" which was presented at the 1973 National ACM Conference and which is also available as a technical report from the Cambridge Project. Janus is also described extensively in previous reports submitted by the Cambridge Project.

There are currently two versions of Janus, a prototype, which has been available to users for over two years, and Version 1, which is currently under development. Almost all work over the last half year has been on the new version of Janus. At the beginning of this time most of the design and many of the basic routines were complete. Now there are over twice as many routines completed, three of the basic data-editing commands are working, the substructure of the system is working and reasonably stable, and another three major commands are nearing completion (define_attribute, create_attribute, and compute). The new version of Janus will be available soon for some users, and will replace the Prototype in the spring of 1974.

The prototype Janus has changed very little recently, but has had considerable use. On the M.I.T. Multics system more than five sessions per day are logged; in addition, there is a copy of it being used heavily on a Multics system at the Pentagon for a variety of tasks ranging from resource allocation to survey analysis.

3.2 SURVEIR Data Management Project

Principals: Michael Ross, Wendy Mela

The Project's effort in the last six months has involved the conversion of the raw data input (i.e., the Multipunched Survey Responses) into JANUS format and adding the relevant information to the Data-Descriptor files. This program reads the Data-Descriptor file produced by the codebook conversion and column-binary tape files (these files are sent to us from the

Preceding page blank

Roper Center's RCA machine and unfortunately lead to serious technical problems on the IBM 370), and generates a Janus dataset for Multics and a new Data-Descriptor File for the retrieval programs (SURVEIR) on Multics. This updated data descriptor file contains data-dependent information, such as the existence of multipunched data and the frequency for each variable, and basic statistics.

A subset of these surveys has been processed and is being tested on SURVEIR. Refinements are being made in the codebook and response conversion programs in order to handle more complex questionnaire formats. By the end of the project, we plan to demonstrate a larger subset (approximately 26 surveys) to a group of social scientists who rely heavily upon secondary analysis of public opinion data in order to assess the adequacy of the system, and the prospects for expanding the database, and offering a user service as a part of the Cambridge Project.

4.0 DATA ANALYSIS

4.1 Project Diana

Principal: Rosemarie Rogers

Contributors: Elaine C. Franklin, Edward J. McCabe,
Kathleen M. O'Connell

Special contributor: John C. Klensin

The major part of the work on Project Diana during the past half year has been devoted to programming. The work of programming the Thesaurus Manager is complete and the programs are part of the Consistent System. The thesaurus that we have designed and implemented is basically a hierarchical structure in which concepts are related in a parent-child fashion. The thesaurus provides for synonym recognition. Terms entered in the thesaurus may be single words or word phrases. The two programs for creating the on-line thesaurus are: `init_diana_tables`, and `build_thesaurus`. Five programs have been written which permit easy updating of the thesaurus: `add_concept`; `add_synonym`; `delete_concept`; `delete_synonym`; and `move_subtree`. These programs will be most useful once the actual document encoding and experimenting has begun, since they will allow us to modify the thesaurus as deemed necessary in light of increased knowledge and data. We have also written a number of thesaurus utilities for easy inspection of the thesaurus contents.

The program package for text and query analysis and document retrieval is also complete. It consists of newly written programs and other programs which are part of our system for semiautomatic thesaurus generation (the "clustering" package) and which are now also included in the actual text manipulation package. A program for creating the negative dictionary (`create_neg_dict`) was written. The package for text analysis and retrieval consists of programs for processing text against the negative dictionary (`run_neg_dict`), thesaurus look-up programs, the suffix removal program, programs for matching document vectors against query vectors and for outputting the references to pertinent documents. Once we have had a chance to check our algorithms in an actual experimental environment, these last processing programs may be modified to incorporate the results of our experimentation.

A report on the Diana system as it currently exists is now being written by Rosemarie Rogers and Elaine C. Franklin. It

will describe in detail the programs just mentioned as well as the data on which the system is built (the negative dictionary, the thesaurus for American documents and the thesaurus for Soviet documents). The report is expected to appear in March; prepublication copies will be available in early February 1974.

In addition to the efforts just described, we have continued to integrate into the Consistent System those programs which were written for Project Diana before all the guidelines for programming for the Consistent System had been delineated. The Diana programs stem text and merge (from the clustering package) are now part of the Consistent System.

We have also continued our analysis of clustering experiments performed in late 1972 and in 1973, and are currently designing a further set of such experiments. When these are completed and analyzed, a report on the entire work will be written by Rosemarie Rogers. Data obtained in the earlier experiments have been incorporated into the thesaurus for American documents, and a preliminary version of the report has been written.

4.2 TSP Project

Principal: John Brode

Contributor: Paul Werbos

As described in the last semiannual report, TSP is now fully operational on Multics within the Consistent System. Automatic access from TSP to Janus or Consistent System files is included so that the user can use data that is stored elsewhere. In addition, the user can now access Multics files or tapes so that data from other machines and sources can be readily brought into TSP and then stored permanently in a Janus dataset or in Consistent System mnarray files.

There are more than 30 matrix commands in addition to this list of the available statistical functions.

<u>TSP Name</u>	<u>Purpose</u>
arma	Autoregressive moving-average estimation (Box-Jenkins)
armawt	Same as arma but with correction for time trend (exponential)

bayes	Bayesian regression
capital	Estimation of capital stock with depreciation
compare	Theil statistics for the comparison of time series (with plot)
constant	Defines constant terms for nonlinear estimation or models
corc	Corchrane-Orcutt estimation for autoregressive error term
correl	Correlation matrix
covar	Covariance matrix
csmean	Mean of several attributes for each entity
dot	Stores series of suffixes
eigen	Eigenvalues of a square matrix
extrap	Bayesian extrapolation
factor	Factor analysis (varimax rotation)
fastfc	Inverse fast Fourier transform
fastfr	Fast Fourier transform
fit	Extrapolation of regression results with Theil statistics and plot
formcp	Forms "cp" matrix (correlation or covariance matrix) for olscp
frml	Stores Polish string representation of a formula
graph	Produces an offline graph
hilu	Hildreth-Liu estimation for autoregressive error term
inst	Instrumental variable estimation (two stage least squares)
interp	Creates a regular time series using data from

irregular intervals

lsq	Nonlinear least squares estimation
namecp	Stores names of attributes in a "cp" matrix
ols	Least Squares (one-pass estimation)
olscp	Least Squares from a "cp" matrix
olsq	Least Squares (two-pass estimate using ortho-normalization - Gram-Schmidt)
olsqwt	Least Squares weighted (two-pass)
olswt	Least Squares weighted (one pass)
param	Defines parameters for nonlinear estimation or models
parm	Posterior distribution of linear combinations of coefficients
pdl	Distributed lag estimation using polynomial interpolation
plot	Online scatter graph
prcomp	Principal components
prob	Posterior probabilities of linear combinations of coefficients
random	Returns a random attribute distributed $N(0,1)$
resid	Bayesian residual analysis
signt	Nonparametric sign test
solve	Solves a system of linear equations
spectr	Spectral analysis
srnk	Spearman rank correlation test
tscorc	Two-stage Corchrane-Orcutt estimation
tshilu	Two-stage Hildreth-Liu estimation

tsplot	Linear plot (on-line) of time series
usecp	Sets default name for "cp" matrix
user	Doorway to user-written subroutines
utest	Mann-Whitney U-test
varian	Analysis of variance

4.3 DATATRAN

Principal: John Brode

Contributors: Jeffrey Stamen, Robert Wallace

The DATATRAN language has been conceived as a means of separating the user from the problems of passing from one program or system to another. Within DATATRAN, all statistical programs and all matrix operations will be considered as single or multivalued functions. The user will be able to nest these functions as well as include them in an arbitrary algebraic expression. The user will not have to be concerned with whether all of the functions being used are within one or another package. Any transferring of control or data will be done implicitly without user attention.

DATATRAN deals with the expression found to the right of the equal-sign (or replacement operator in APL). There is no limit to the length and complexity of expressions. Subexpressions can be nested within expressions by using parentheses. For example:

```
stdev(log(5+score/mean(score)))
```

When calling a function, the user can specify whether no print, some print, or all print is desired. For example:

```
regress(residuals of regress (A on X Y Z with print all) on X Y Z  
with print some)
```

On the inner call to regress, all results are printed but on the outer call only partial results appear at the console. The user can specify leads and lags by putting "t" plus or minus the number of periods to be led or lagged in parentheses after the attribute name. For instance:

```
regress(A on X(t-1) Y(t-1) Z(t-1))
```

gets a regression explaining A by the values of X, Y, and Z for the previous period.

It is further possible to save any of the results of a call to a function by using a save phrase in the call to the function. A save phrase, at its simplest, is the word "save" followed by a key word (different for every function) that indicates what is to be saved as an attribute, number, or matrix as in the following example:

```
cra independent data as matrix(log(data),time)
cm matrix_mult(inverse      (cross_products(independent_data)
  save inverse_xx)matrix_mult (transpose(independent_data)
  log(dependent_attribute)))
```

As of the end of the year, the DATATRAN language has been specified. The bulk of the coding to implement this language has been finished and elementary expressions can be run. It is estimated that the entire language system will be operational by the end of April 1974.

4.4 Economic Application of Spectral Analysis

Principal: Robert F. Engle

Introduction

This project began in September 1971 and has just now been completed in December 1973. The overall goals were to program facilities for performing spectral analyses of particular interest to economists, evaluate these procedures in the context of economic data and problems, and explore possible empirical applications as only in this way can the usefulness of the procedures be fully documented. This report will discuss work along each of these lines.

In summary, the programs are complete and well documented. Monte Carlo studies indicated that the small sample properties of these estimators are better than often anticipated, and applications of these techniques to estimation of investment and consumption functions indicate the promise of spectral methods in economics.

Programs

The programs are a set of subroutines which in combination

compute spectral and cross-spectral estimates, or in connection with a regression procedure, compute efficient estimates of a linear regression with serial correlation (the Hannan estimator), or band spectrum regressions. These subroutines are all called by a master subroutine complete with defaults so that the user need know nothing about spectral analysis in order to obtain the analysis. Yet, the subroutines are flexible so that a sophisticated user can generate his own techniques.

The user interface as well as the component parts are described in a manual to be available soon. This manual contains a careful introduction to the interpretation and understanding of the tools. Coupled with this is a more advanced paper, Engle (2), describing the relation between spectral methods and time domain methods (distributed lag and Box-Jenkins procedures) as well as the justification for the procedures used in the spectral computations. The paper is still, however, a tutorial piece, and can be read by a user without any knowledge of spectral analysis.

Testing of the Estimators

The desirability of correcting for serial correlation in a linear regression, in order to obtain efficient estimates, is widely recognized. However, when a specific form for this serial correlation is not known, the principle estimator which can guarantee asymptotic efficiency is a spectral estimator, originally proposed by Hannan. It is rarely used, however, partly because the programming is difficult and partly because it presumably has far worse finite sample behavior than asymptotic behavior.

Since this estimator is so desirable theoretically, it was included in the spectral package. In order to examine the finite sample properties of the estimator, Roy E. Gardner from Cornell University and Robert F. Engle performed an extensive Monte Carlo test in a variety of economically relevant environments.

The conclusions described in Engle and Gardner (4) indicate that when serial correlation is of a relatively simple form, then the Hannan estimator is virtually equivalent for samples of size 50 or more. When serial correlation is more complicated and severe, the Hannan estimator has roughly twice its asymptotic variance at a sample of size 100. In all cases, the spectral estimators appeared to be unbiased for sample sizes over 50. There are many qualifications to these results described in the original paper but the conclusions are clear: that for economically relevant sample sizes (US data post-WWII; quarterly or monthly data) these estimators are quite well-behaved.

Economic Applications

A paper on band spectrum regression, Engle (1), describes sensible economic applications of spectral analysis to problems of estimating regressions subject to seasonality, errors in variables, or misspecification in different frequency bands. A test statistic is derived for testing the hypothesis that the regression coefficients are not stable across frequency bands.

In this paper, an application to the estimation of consumption functions is examined. If the permanent income hypothesis is valid in its simplest form, then the marginal propensity to consume out of transitory income should be smaller than out of permanent income. Identifying the high frequency components as transitory income, we should reject the hypothesis that the regression coefficients are the same at different frequencies. In fact, we are not able to reject this hypothesis and thus, at least in this simple model, the permanent income hypothesis receives no support.

In a second application of these techniques, an investment function was estimated by Duncan K. Foley and R.F. Engle.(3) This function was derived from a theory of macroeconomic behavior based upon supply limitations. The data, however, suffered from errors of observation which were assumed to lie within particular frequency bands. The model was estimated using the technique of band spectrum regression with remarkable success, suggesting the usefulness of spectral techniques for macroeconomic applications.

Bibliography:

1. Engle, Robert F., "Band Spectrum Regression", International Economic Review, 1974.
2. Engle, Robert F., "An Introduction to the Spectral Analysis Capability", mimeo.
3. Engle, Robert F., and Duncan K. Foley, "An Asset Price Model of Aggregate Investment", submitted for publication.
4. Engle, Robert F., and Roy Gardner, "Some Finite Sample Properties of Spectral Estimators of a Linear Regression", M.I.T. Working Paper No.122, December 1973.

5.0 MODELING

5.1 The General Implicator

Principal: Ithiel de Sola Pool

Contributor: Shahriar Ahy

The semiannual report of January-June 1973 discussed the purpose and structure of the General Implicator and the status at that time. Since that report was written, a technical paper "Text Representation, Text-Data Management, and Text Modeling with the General Implicator" (Ahy, Shahriar and Ithiel de Sola Pool, The Cambridge Project, M.I.T. Sept. 1973) has been produced in draft form and is in the process of final draft for publication. The paper covers, rather completely, these four topical areas:

1. A statement on the purpose of the General Implicator Project.
2. An outline of how the General Implicator looks in its first implementation in the Cambridge Project Consistent System on the Multics host computer.
3. A theoretical statement of the relation between the General Implicator's representation of English statements and those formal representations of meaning that have proved useful in other artificial intelligence research. This section represents our second thoughts, or, in other words, the insights we have acquired in doing the first version of the General Implicator.
4. A brief statement of next steps in the research.

5.2 Discourse

Principal: Wren McMains

Urban planners and designers work in a tentative and exploratory way. They describe the environment into which they intend to intervene. They transform, hypothetically of course, that environment because of certain attitudes they have developed toward it or because of the attitudes of others. They display the transformed environment to examine it and they often test the transformed environment according to criteria which they have developed to measure its performance. Often after one such

cycle, the planner displays the results of his efforts to groups with whom he is working, trying to make them understand the alternatives he has created, and how well they perform, as well as to discuss the characterization of the problem which his alternatives and tests imply. Frequently the results of this examination may result in the problem being formulated anew and in the description being redone. In order to improve the description or to change it the need arises for new sources of data from which conclusions can be drawn.

The Discourse language has been developed for the designer and planner as a tool which could keep pace with his tentative and exploratory way of working. It provides the designer with ways of describing environments as he chooses rather than in providing specific types of descriptions. However, since location plays such an important role in most environment descriptions, it has become a descriptor which Discourse allows to appear implicitly in other descriptions. In addition to its assistance in locational descriptions, Discourse provides a means by which the designer can describe his own transformations, can specify his own displays, or can test the environment in the ways of his own choosing rather than in suggesting specific types of transformations, rigidifying the displays, or prespecifying environmental performance criteria. Additionally, Discourse is designed to be convenient to use and in keeping with the designer's thinking and conventional modes of expression.

The last six months have been devoted to documenting the operation of Discourse commands. Since adequate help was never found, this has been a part-time effort. The new reference manual is not complete, but parts of it are available; in particular, descriptions of commands that have been added since early 1972. Reprints of parts of the old manual that cover the other commands are being made available in a consistent form, but remain to be updated to reflect current operation.

The Discourse-IMLAC communicator was completed. It allows a Discourse user, utilizing an IMLAC terminal, to see a continually up-to-date display of several attributes while he gives Discourse commands just as if he were using a nongraphic console. This also prevents interrupting his train of thought just to issue "map" or other display commands. The communicator also provides graphic input of both locational and numerical information.

Several new commands were added during the period; however, the development was funded by other sources. The most interesting of these being:

allocate	Distributes one set of charvar (CHARacteristics which VARY with location) values to another set of locations (e.g., employment to residential locations)
accessibility	Allows testing accessibility to various kinds of services as well as activity clustering
map_values	Maps charvar values (the map command maps attribute locations)
categorize	Provides another way to look at charvar values (A special feature also allows it to be used to produce color maps, either with the School of Architecture and Planning color scope or on a bi-color offset printing from plates prepared on a IMLAC.)
separation	In addition to being a global version of the basic nearest command, provides for mean and maximum separations as well as moments
store_state	Together with "restore_state", allows for fast switching between various design states. In addition, the restore_state program curial provides a way to return to the state Discourse was in when a Multics crash occurred; however, since this feature results in additional storage costs it must be explicitly requested by the user.

For the Discourse that becomes a permanent part of the Consistent System, the remaining tasks are to be completed:

- (1) Update any old command descriptions that do not correctly reflect present operation.
- (2) Documentation of internal operation, at least to the point that new programmers would have sufficient information to add new commands.
- (3) Minor improvement in the interface between Discourse and other Consistent System programs and systems. Present communication is between Discourse arrays and Consistent System mnarrays. Users have found this awkward when they want to have Consistent System routines operate on information stored in charvars, since they must first format it into arrays. We now have enough experience with the

steps they most often do to make the interface as transparent as possible; this, we think, is important if users are to work freely in the Consistent System.

6.0 HUMAN FACTORS

6.1 A Modular Computer System for the Study of Autonomic Behavior

Principal: Craig Fields

This project has been described in detail in our previous reports. We wish to report that this project has merited funding, separately from the Cambridge Project, under its own contract. It is rewarding that progress has been made in this endeavor and we (the Cambridge Project) are pleased to have been involved in it.

6.2 Simulation Gaming as a Behavioral Laboratory

Principal: Peter G. W. Keen

During the summer, the large-scale management simulation game, developed at the Harvard Business School by a research team which included P. G. W. Keen, was documented as the first step towards implementing it on Multics. Since that time several improvements have been made in the game (at Harvard); these changes will be incorporated into the documentation and it is planned to begin conversion to Multics by the late spring. Funds are being requested from several outside sources; the full implementation of the game will require about an 18-month effort and it is planned to coordinate this effort at M.I.T. Sloan School, using technical support from the Cambridge Project.

When the game is operational, it will be made available to other colleges and used as a basis for comparative experiments on the impact of information structure and systems on participant behavior, problem-solving strategies, and the use of analytic methods and models to support decision-making.

As part of the documentation effort during the summer, this researcher also began several working papers describing both the game, its use, and its implications as a behavioral laboratory.

Final drafts of these papers will be made available around March 1, 1974.

6.3 IMLAC Graphics

Principals: Nicholas Negroponte, M. S. Miller

A complete system has been implemented which installs and maintains on the Architecture Machine facility (M.I.T. Department of Architecture) IMLAC graphic terminal core images. Upon request, these images may be transmitted over switched telephone lines at 1200 baud to bootstrap the terminals. Components of the system include:

1. An Architecture Machine program to read and file a paper tape containing the "block loader" that always prefaces an actual core image.
2. An Architecture Machine program to read and file a paper tape containing the core image of an IMLAC program. This tape must be in standard "block absolute" format with interrecord gaps consisting of at least two null tape characters.
3. An Architecture Machine program to monitor the half-duplex telephone interface (connected to a Bell 202C modem) and transmit the block loader followed by the core image requested. An incoming ring is automatically answered, and "handshaking" protocols are initiated to raise the carrier signal so that primary data may be sent to the Architecture Machine and supervisory data may be sent from it. A character string identifying the core image is read with standard provision for "character erase" and "line kill". When the string is terminated (by receipt of the space character), the data directions are reversed, and transmission to the terminal begins. When it is complete, hangup occurs and the program awaits another request. All errors (failure to establish carrier, file not found, disconnect, etc.) cause hangup.
4. An IMLAC program to monitor the keyboard, transmit a typed character string and, upon transmission of the space character, read into high core the block loader and transfer control to it. This program is implemented in 32 instructions and data words as a Read Only Memory (ROM) so that a programmer console is not necessary to repair the program in case of accidental damage.
5. An IMLAC program to read a core image and transfer control to it upon completion. This is the block loader. As the core image is read in, checksums are computed to verify correct transmission, although the current block loader simply halts if an error is detected.

The system has been thoroughly tested with the Cambridge Project IMLAC located at The Architecture Machine facility and works flawlessly. Successful bootstrapping of another terminal at 575 Technology Square has also been accomplished. Other attempts at Technology Square and at Harvard have failed due to faulty modems, ROMs, or asynchronous interfaces -- all at the terminal end. Clearly, each terminal system must be checked out and made to function correctly.